Whitepaper zu AWS

Übersicht über die Sicherheit von AWS Lambda

Übersicht über die Sicherheit von AWS Lambda: Whitepaper zu AWS

Table of Contents

Übersicht über die Sicherheit von AWS Lambda

Veröffentlichungsdatum: 12. Februar 2021 (Dokumentversionen)

Überblick

Dieses Whitepaper stellt einen tiefen Einblick in den AWS Lambda-Service durch eine Sicherheitslinse dar. Es stellt einen guten Überblick des Service dar, der für neue Anwender nützlich sein kann und detailliertere Informationen zu Lambda für aktuelle Benutzer bietet.

Die Zielgruppe für dieses Whitepaper sind Chief Information Security Officers (CISOs), Informationssicherheitstechniker, Unternehmensarchitekten, Compliance-Teams und alle anderen, die daran interessiert sind, die Grundlagen von AWS Lambda zu verstehen.

Einführung

Heutzutage verwenden mehr Workloads AWS Lambda, um Skalierbarkeit, Leistung und Kosteneffizienz zu erreichen, ohne die zugrunde liegende Infrastruktur zu verwalten. Diese Workloads werden auf Tausende von gleichzeitigen Anforderungen pro Sekunde skaliert. Lambda ist einer der vielen wichtigen Dienste, die heute von AWS angeboten werden. Lambda wird von Hunderttausenden Amazon Web Services (AWS)-Kunden verwendet, um jeden Monat Billionen von Anfragen zu verarbeiten.

Lambda eignet sich für unternehmenskritische Anwendungen in vielen Branchen. Eine Vielzahl von Kunden, die von Medien und Unterhaltung bis hin zu Finanzdienstleistungen und anderen regulierten Branchen reichen, nutzen Lambda. Diese Kunden verkürzen die Markteinführungszeit, optimieren die Kosten und verbessern die Agilität, indem sie sich auf das konzentrieren, was sie am besten können: ihr Geschäft führen.

Das Modell der verwalteten Laufzeitumgebung ermöglicht es Lambda, viele der Implementierungsdetails der Ausführung von Serverless-Workloads zu verwalten. Dieses Modell verringert die Angriffsfläche noch mehr und vereinfacht gleichzeitig die Cloud-Sicherheit. Dieses Whitepaper stellt Entwicklern, Sicherheitsanalysten, Sicherheits- und Compliance-Teams sowie anderen Interessensgruppen die Grundlagen dieses Modells zusammen mit bewährten Methoden vor.

Informationen zu AWS Lambda

AWS Lambda ist ein ereignisgesteuerter, serverloser Computing-Service, der andere AWS-Services mit benutzerdefinierter Logik erweitert oder andere Backend-Services bereitstellt, die mit Skalierung, Leistung und Sicherheit arbeiten. Lambda kann Code automatisch als Reaktion auf mehrere Ereignisse wie HTTP-Anforderungen über Amazon API Gateway, Änderungen von Objekten in Amazon S3-Buckets, Aktualisierungen von Tabellen in Amazon DynamoDB oder Zustandsübergänge in AWS Step Functions ausführen. Sie können Code auch direkt von jeder Web- oder mobilen App aus ausführen. Lambda führt Code auf einer hoch verfügbaren Datenverarbeitungsinfrastruktur aus und erledigt die gesamte Administration der zugrunde liegenden Plattform, einschließlich der Server- und Betriebssystemwartung, Kapazitätsbereitstellung und automatischen Skalierung, Patching, Code-Überwachung und -Protokollierung.

Mit Lambda können Sie Ihren Code einfach hochladen und konfigurieren, wann er aufgerufen werden soll. Lambda übernimmt alles, was zum Ausführen Ihres Codes für hohe Verfügbarkeit erforderlich ist. Lambda lässt sich in viele andere AWS-Services integrieren und ermöglicht es Ihnen, Serverless-Anwendungen oder Backend-Services zu erstellen, die von regelmäßig ausgelösten, einfachen Automatisierungsaufgaben bis hin zu vollwertigen Microservice-Anwendungen reichen.

Lambda kann auch so konfiguriert werden, dass es auf Ressourcen in Ihrer Amazon Virtual Private Cloud und damit auf Ihre lokalen Ressourcen zugreift.

Mit AWS Identity and Access Management (IAM) und anderen in diesem Whitepaper behandelten Techniken können Sie Lambda problemlos mit einem starken Sicherheitsstatus bereitstellen, um ein hohes Maß an Sicherheit und Überprüfung zu gewährleisten und Ihre Compliance-Anforderungen zu erfüllen.

Themen

Vorteile von Lambda

Kunden, die die Kreativität und Geschwindigkeit ihrer Entwicklungsorganisationen vorantreiben möchten, ohne die Fähigkeit ihres IT-Teams zu beeinträchtigen, eine skalierbare, kostengünstige und verwaltbare Infrastruktur bereitzustellen, schätzen, dass sie mithilfe von AWS Lambda betriebliche

Komplexität gegen Agilität und bessere Preise eintauschen können – ohne Einbußen bei Skalierung oder Zuverlässigkeit.

Lambda bietet viele Vorteile, darunter die folgenden:

Keine zu verwaltenden Server.

Lambda führt Ihren Code auf einer hochverfügbaren, fehlertoleranten Infrastruktur aus, die über mehrere Availability Zones (AZs) in einer einzigen Region verteilt ist. Dabei wird Code nahtlos bereitgestellt und die gesamte Verwaltung, Wartung und Patches der Infrastruktur geboten. Lambda bietet auch integrierte Protokollierung und Überwachung, einschließlich der Integration mit Amazon CloudWatch, CloudWatch Logs und AWS CloudTrail.

Kontinuierliche Skalierung

Lambda verwaltet die Skalierung Ihrer Funktionen (oder Anwendungen) präzise, indem es ereignisgesteuerten Code parallel ausführt und jedes Ereignis einzeln verarbeitet.

Millisekunden-Messung

Mit AWS Lambda wird Ihnen jede 1-Millisekunden (ms)-Einheit, in der Ihr Code ausgeführt wird, sowie die Anzahl der Ausführungen des Codes in Rechnung gestellt. Sie zahlen für einen konstanten Durchsatz oder eine konstante Ausführungsdauer anstatt für eine Servereinheit.

Erhöht die Innovation

Lambda setzt Ihre Programmierressourcen frei, indem es das Infrastrukturmanagement übernimmt, sodass sich dieses stärker auf Innovation und Entwicklung der Geschäftslogik konzentrieren kann.

Modernisieren Sie Ihre Anwendungen

Mit Lambda können Sie Funktionen mit vorab trainierten Modellen für Machine Learning verwenden, um künstliche Intelligenz einfach in Anwendungen zu integrieren. Eine einzige Anforderung der Programmierschnittstelle (API) kann Bilder klassifizieren, Videos analysieren, Sprache in Text umwandeln, die Verarbeitung natürlicher Sprache durchführen und vieles mehr.

Umfassendes Ökosystem

Lambda unterstützt Entwickler über das AWS Serverless Application Repository für die Erkennung, Bereitstellung und Veröffentlichung von serverlosen Anwendungen oder das AWS Serverless

Application Model zum Erstellen von serverlosen Anwendungen und Integrationen mit verschiedenen integrierten Entwicklungsumgebungen (IDEs) wie AWS Cloud9AWS Toolkit for Visual Studio, AWS Tools for Visual Studio Team Services und viele andere. Lambda ist in zusätzliche AWS-Services integriert, um Ihnen ein reichhaltiges Ökosystem für die Erstellung von serverlosen Anwendungen zu bieten.

Kosten für die Ausführung von Lambda-basierten Anwendungen

Lambda bietet ein präzises nutzungsbasiertes Preismodell. Hier erfolgt die Abrechnung basierend auf der Anzahl der Funktionsaufrufe und ihrer Dauer (der Zeit, die der Code benötigt, um ausgeführt zu werden). Zusätzlich zu diesem flexiblen Preismodell bietet Lambda auch 1 Million unbefristete kostenlose Anfragen pro Monat an, sodass viele Kunden ihren Prozess ohne Kosten automatisieren können.

Das Modell der übergreifenden Verantwortlichkeit

Sicherheit und Compliance stellen eine geteilte Verantwortlichkeit zwischen AWS und dem Kunden dar. Dieses Modell der geteilten Verantwortlichkeit bedeutet für Sie eine Erleichterung des Betriebs, da AWS die Komponenten des Host-Betriebssystems und der Virtualisierungsebene betreibt, verwaltet und steuert und zudem für die physische Sicherheit der Standorte sorgt, an denen die Services ausgeführt werden.

Für AWS Lambda verwaltet AWS die zugrunde liegende Infrastruktur und grundlegende Services, das Betriebssystem und die Anwendungsplattform. Sie sind für die Sicherheit Ihres Codes und das Identity and Access Management (IAM) zum Lambda-Service und innerhalb Ihrer Funktion verantwortlich.

Abbildung 1 zeigt das Modell der geteilten Verantwortung für die gemeinsamen und unterschiedlichen Komponenten von AWS Lambda. AWS-Zuständigkeiten werden unter der gepunkteten Linie orange angezeigt und die Zuständigkeiten des Kunden werden über der gepunkteten Linie blau angezeigt.

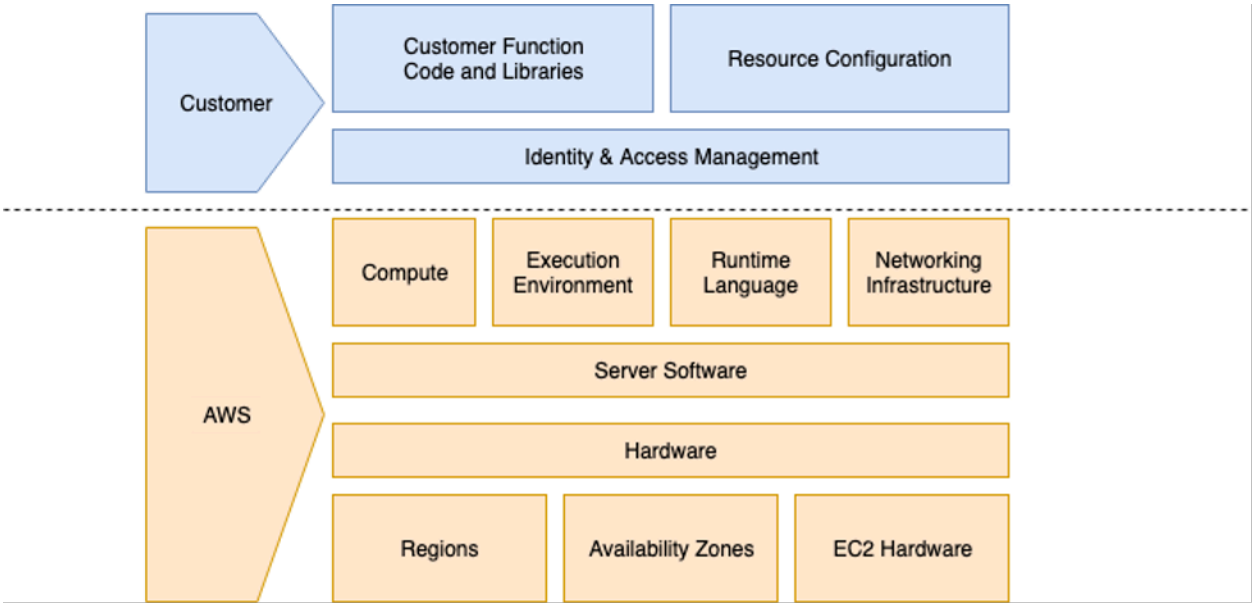

Abbildung 1 – Modell der geteilten Verantwortung für AWS Lambda

Lambda-Funktionen und -Ebenen

Mit Lambda können Sie Code virtuell ohne Verwaltung der zugrunde liegenden Infrastruktur ausführen. Sie sind nur für den Code verantwortlich, den Sie Lambda bereitstellen, und für die Konfiguration, wie Lambda diesen Code in Ihrem Namen ausführt. Heute unterstützt Lambda zwei Arten von Code-Ressourcen: Funktionen und Ebenen.

Eine Funktion ist eine Ressource, die aufgerufen werden kann, um Ihren Code in Lambda auszuführen. Funktionen können eine gemeinsame oder geteilte Ressource namens Ebenen enthalten. Ebenen können verwendet werden, um gemeinsamen Code oder Daten über verschiedene Funktionen oder AWS-Konten hinweg gemeinsam zu nutzen. Sie sind für die Verwaltung des gesamten in Ihren Funktionen oder Ebenen enthaltenen Codes verantwortlich. Wenn Lambda den Funktions- oder Ebenencode von einem Kunden erhält, schützt Lambda den Zugriff darauf, indem es ihn im Speicher mit AWS Key Management Service (AWS KMS) und während der Übertragung mithilfe von TLS 1.2+ verschlüsselt.

Sie können den Zugriff auf Ihre Funktionen und Ebenen über AWS-Lambda-Richtlinien oder ressourcenbasierte Berechtigungen verwalten. Eine vollständige Liste der unterstützten IAM-Funktionen auf IAM finden Sie unter IAM-kompatible AWS-Services.

Sie können auch den gesamten Lebenszyklus Ihrer Funktionen und Ebenen über die Steuerungsebenen-APIs von Lambda steuern. Sie können beispielsweise Ihre Funktion löschen, indem Sie `DeleteFunction` aufrufen, oder Berechtigungen für ein anderes Konto widerrufen, indem Sie `RemovePermission` aufrufen.

Lambda-Aufrufmodi

Die Aufruf-API kann in zwei Modi aufgerufen werden: Ereignismodus und Request-Response-Modus.

- Im Ereignismodus wird die Nutzlast für einen asynchronen Aufruf in die Warteschlange gestellt.
- Der Request-Response-Modus ruft die Funktion synchron mit der bereitgestellten Nutzlast auf und gibt sofort eine Antwort zurück.

In beiden Fällen wird die Funktionsausführung immer in einer Lambda-Ausführungsumgebung ausgeführt, aber die Nutzlast nimmt unterschiedliche Pfade. Weitere Informationen finden Sie unter „Lambda-Ausführungsumgebungen“ in diesem Dokument.

Sie können auch andere AWS-Services verwenden, die in Ihrem Namen Aufrufe ausführen. Welcher Aufrufmodus verwendet wird, hängt davon ab, welchen AWS-Service Sie verwenden und wie er konfiguriert ist. Weitere Informationen zur Integration anderer AWS-Services in Lambda finden Sie unter Verwenden von AWS Lambda mit anderen Services.

Wenn Lambda einen Request-Response-Aufruf erhält, wird er direkt an den Aufrufservice übergeben. Wenn der Aufrufservice nicht verfügbar ist, können Aufrufende die Nutzlast vorübergehend clientseitig in eine Warteschlange stellen, um den Aufruf eine festgelegte Anzahl von Malen zu wiederholen. Wenn der Aufrufservice die Nutzlast empfängt, versucht der Dienst dann, eine verfügbare Ausführungsumgebung für die Anforderung zu identifizieren, und übergibt die Nutzlast an diese Ausführungsumgebung, um den Aufruf abzuschließen. Wenn keine vorhandenen oder geeigneten Ausführungsumgebungen vorhanden sind, wird eine dynamisch als Antwort auf die Anforderung erstellt. Während der Übertragung werden an den Aufrufservice gesendete Aufrufnutzlasten mit TLS 1.2+ gesichert. Der Datenverkehr innerhalb des Lambda-Service (von der Lastenverteilung abwärts) durchläuft eine isolierte interne virtuelle private Cloud (VPC), die dem Lambda-Service gehört, innerhalb der AWS-Region, an die die Anforderung gesendet wurde.

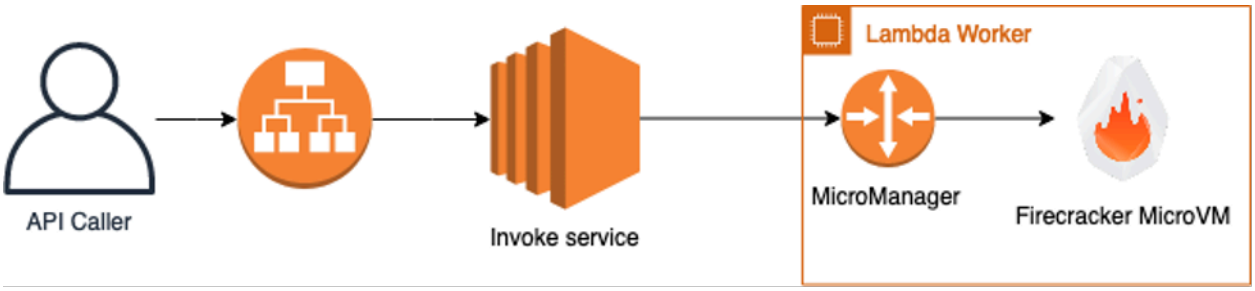

Abbildung 2 – Aufrufmodell für AWS LambdaRequest-Response.

Nutzlasten des Ereignisaufruf-Modus werden vor dem Aufruf immer zur Verarbeitung in die Warteschlange gestellt. Alle Nutzlasten werden für die Verarbeitung von Amazon Simple Queue Service (Amazon SQS) in die Warteschlange gestellt. Ereignisse in der Warteschlange sind während der Übertragung immer mit TLS 1.2+ gesichert, aber sie werden derzeit im Speicher nicht verschlüsselt. Die von Lambda verwendeten Amazon SQS-Warteschlangen werden vom Lambda-Service verwaltet und sind für Sie als Kunde nicht sichtbar. Ereignisse in der Warteschlange können in einer gemeinsam genutzten Warteschlange gespeichert werden, können jedoch migriert oder dedizierten Warteschlangen zugewiesen werden, abhängig von einer Reihe von Faktoren, die nicht direkt von Kunden gesteuert werden können (z. B. Aufrufrate, Ereignisgröße usw.).

Ereignisse in der Warteschlange werden von der Poller-Flotte von Lambda stapelweise abgerufen. Die Poller-Flotte ist eine Gruppe von EC2-Instances, deren Zweck es ist, Ereignisaufrufe in der Warteschlange zu verarbeiten, die noch nicht verarbeitet wurden. Wenn die Poller-Flotte ein Ereignis aus der Warteschlange abruft, das sie verarbeiten muss, übergibt sie es an den Aufrufservice, genau wie ein Kunde es in einem Request-Response-Modus aufrufen würde.

Wenn der Aufruf nicht ausgeführt werden kann, speichert die Poller-Flotte das Ereignis vorübergehend im Speicher auf dem Host, bis sie entweder die Ausführung erfolgreich abschließen kann oder bis die Anzahl der Wiederholungsversuche überschritten wurde. In der Nutzlastflotte selbst werden niemals Nutzdaten auf die Festplatte geschrieben. Die Abfrageflotte kann über AWS-Kunden hinweg beauftragt werden, um die kürzeste Zeit bis zum Aufruf zu ermöglichen. Weitere Informationen darüber, welche Dienste den Ereignisaufrufmodus verwenden können, finden Sie unter Verwenden von AWS Lambda mit anderen Services.

Lambda-Ausführungen

Wenn Lambda eine Funktion in Ihrem Namen ausführt, verwaltet es sowohl die Bereitstellung als auch die Konfiguration der zugrunde liegenden Systeme, die zur Ausführung Ihres Codes erforderlich sind. Auf diese Weise können sich Ihre Entwickler auf die Geschäftslogik und das Schreiben von Code konzentrieren, ohne die zugrunde liegenden Systeme verwalten zu müssen.

Der Lambda-Service ist in die Steuerungsebene und die Datenebene aufgeteilt. Jede Ebene dient einem bestimmten Zweck im Service. Die Steuerungsebene stellt die Verwaltungs-APIs bereit (z. B. `CreateFunction`, `UpdateFunctionCode`, `PublishLayerVersion` usw.) und verwaltet Integrationen mit allen AWS-Services. Die Kommunikation mit der Steuerebene von Lambda ist während der Übertragung durch TLS geschützt. Alle Kundendaten, die in der Steuerebene von Lambda gespeichert sind, werden im Ruhezustand mithilfe von AWS KMS verschlüsselt, um sie vor unbefugter Offenlegung oder Manipulation zu schützen.

Die Datenebene ist Lambdas Invoke-API, die den Aufruf von Lambda-Funktionen auslöst. Wenn eine Lambda-Funktion aufgerufen wird, weist die Datenebene dieser Funktionsversion eine Ausführungsumgebung auf einem AWS-Lambda-Worker (oder einfach Worker – eine Art Amazon-EC2-Instance) zu oder wählt eine vorhandene Ausführungsumgebung aus, die bereits für diese Funktionsversion eingerichtet wurde, die es dann verwendet, um den Aufruf abzuschließen. Weitere Informationen finden Sie im Abschnitt „AWS Lambda MicroVMS and Workers“ in diesem Dokument.

Lambda-Ausführungsumgebungen

Jeder Aufruf wird vom Aufrufdienst von Lambda an eine Ausführungsumgebung auf einem Worker weitergeleitet, der die Anforderung bearbeiten kann. Außer über die Datenebene können Kunden und andere Benutzer keine eingehende Netzwerkkommunikation mit einer Ausführungsumgebung direkt initiieren. Auf diese Weise können Sie sicherstellen, dass die Kommunikation mit Ihrer Ausführungsumgebung authentifiziert und autorisiert ist.

Ausführungsumgebungen sind für eine bestimmte Funktionsversion reserviert und können nicht über Funktionsversionen, Funktionen oder AWS-Konten hinweg wiederverwendet werden. Das bedeutet, dass eine einzelne Funktion, die zwei verschiedene Versionen haben kann, zu mindestens zwei eindeutigen Ausführungsumgebungen führen würde.

Jede Ausführungsumgebung kann jeweils nur für einen gleichzeitigen Aufruf verwendet werden, und sie können aus Leistungsgründen für mehrere Aufrufe derselben Funktionsversion wiederverwendet

werden. Abhängig von einer Reihe von Faktoren (z. B. Aufrufrate, Funktionskonfiguration usw.) können eine oder mehrere Ausführungsumgebungen für eine bestimmte Funktionsversion existieren. Mit diesem Ansatz ist Lambda in der Lage, seinen Kunden eine Isolierung auf Funktionsversionsebene bereitzustellen.

Lambda isoliert derzeit keine Aufrufe innerhalb der Ausführungsumgebung einer Funktionsversion. Dies bedeutet, dass ein Aufruf einen Zustand hinterlassen kann, der sich auf den nächsten Aufruf auswirken kann (z. B. in /tmp geschriebene Dateien oder In-Memory-Daten). Wenn Sie sicherstellen möchten, dass ein Aufruf keinen anderen Aufruf beeinflussen kann, empfiehlt Lambda, dass Sie zusätzliche andere Funktionen erstellen. Sie könnten beispielsweise unterschiedliche Funktionen für komplexe Analysevorgänge erstellen, die fehleranfälliger sind, und Funktionen wiederverwenden, die keine sicherheitsempfindlichen Vorgänge ausführen. Lambda begrenzt derzeit nicht die Anzahl der Funktionen, die Kunden erstellen können. Weitere Informationen zu Grenzwerten finden Sie auf der Seite Lambda-Kontingente.

Ausführungsumgebungen werden kontinuierlich von Lambda überwacht und verwaltet und können aus einer Reihe von Gründen erstellt oder zerstört werden, einschließlich, aber nicht beschränkt auf:

- Ein neuer Aufruf tritt auf und es ist keine geeignete Ausführungsumgebung vorhanden.
- Eine interne Laufzeit- oder Worker-Softwarebereitstellung erfolgt.
- Eine neue bereitgestellte Parallelitätskonfiguration wurde veröffentlicht.
- Die Lease-Zeit in der Ausführungsumgebung oder dem Worker nähert sich an die maximale Lebensdauer an oder hat diese überschritten.
- Andere interne Workload-Rebalancing-Prozesse.

Kunden können die Anzahl der vorab bereitgestellten Ausführungsumgebungen verwalten, die für eine Funktionsversion vorhanden sind, indem sie die bereitgestellte Parallelität in ihrer Funktionskonfiguration konfigurieren. Bei entsprechender Konfiguration erstellt, verwaltet und stellt Lambda sicher, dass die konfigurierte Anzahl von Ausführungsumgebungen immer vorhanden ist. Auf diese Weise haben Kunden eine bessere Kontrolle über die Startleistung ihrer Serverless-Anwendungen beliebiger Größe.

Abgesehen von einer bereitgestellten Parallelitätskonfiguration können Kunden die Anzahl der Ausführungsumgebungen, die von Lambda als Reaktion auf Aufrufe erstellt oder verwaltet werden, nicht deterministisch steuern.

Ausführungsrolle

Jede Lambda-Funktion muss auch mit einer Ausführungsrolle konfiguriert werden, bei der es sich um eine IAM-Rolle handelt, die vom Lambda-Service bei der Ausführung von Steuerungs- und Datenebenenoperationen in Bezug auf die Funktion übernommen wird. Der Lambda-Service übernimmt diese Rolle, um temporäre Sicherheitsanmeldeinformationen abzurufen, die dann während des Aufrufs einer Funktion als Umgebungsvariablen verfügbar sind. Aus Leistungsgründen speichert der Lambda-Service diese Anmeldeinformationen im Cache und kann sie in verschiedenen Ausführungsumgebungen wiederverwenden, die dieselbe Ausführungsrolle verwenden.

Um sicherzustellen, dass das Prinzip der geringsten Berechtigung eingehalten wird, empfiehlt Lambda, dass jede Funktion ihre eigene Rolle hat und dass sie mit dem Mindestsatz an Berechtigungen konfiguriert ist, die sie benötigt.

Der Lambda-Service kann auch die Ausführungsrolle übernehmen, um bestimmte Steuerungsebenenoperationen durchzuführen, z. B. solche im Zusammenhang mit dem Erstellen und Konfigurieren von Elastic Network Interfaces (ENI) für VPC-Funktionen und dem Senden von Protokollen an Amazon CloudWatch Application Insights, Senden von Traces an AWS X-Ray oder anderen nicht aufrufbezogenen Vorgängen. Kunden können diese Anwendungsfälle jederzeit ansehen und prüfen, indem sie die Überwachungsprotokolle in AWS CloudTrail ansehen.

Weitere Informationen zu diesem Thema finden Sie auf der Dokumentationsseite der AWS-Lambda-Ausführungsrolle.

Lambda-MicroVMS und Worker

Lambda wird seine Ausführungsumgebungen auf einer Flotte von Amazon-EC2-Instances namens AWS-Lambda-Worker erstellen. Worker sind Bare-Metal-EC2-Nitro-Instances, die von Lambda in einem separaten, isolierten AWS-Konto gestartet und verwaltet werden, das für Kunden nicht sichtbar ist. Worker haben eine oder mehrere Hardware-virtualisierte Micro Virtual Machines (MVM), die von Firecracker erstellt wurden. Firecracker ist ein Open-Source-Virtual-Machine-Monitor (VMM), der die Kernel-basierte virtuelle Maschine (KVM) von Linux zum Erstellen und Verwalten von MVMs verwendet. Es wurde speziell für die Erstellung und Verwaltung sicherer, mehrmandantenfähiger Container und funktionsbasierter Dienste entwickelt, die Serverless-Betriebsmodelle bereitstellen. Weitere Informationen zum Sicherheitsmodell von Firecracker finden Sie auf der Firecracker-Projekt-Website.

Als Teil des Modells der geteilten Verantwortung ist Lambda für die Aufrechterhaltung der Sicherheitskonfiguration, Kontrollen und Patch-Levels der Worker verantwortlich. Das Lambda-Team verwendet Amazon Inspector, um bekannte potenzielle Sicherheitsprobleme sowie andere benutzerdefinierte Mechanismen zur Benachrichtigung über Sicherheitsprobleme und Listen zur Vorabveröffentlichung zu ermitteln, sodass Kunden den zugrunde liegenden Sicherheitsstatus ihrer Ausführungsumgebung nicht verwalten müssen.

Abbildung 3 – Isolationsmodell für AWS Lambda Worker

Worker haben eine maximale Lease-Lebensdauer von 14 Stunden. Wenn sich ein Worker der maximalen Lease-Zeit nähert, werden keine weiteren Aufrufe an ihn weitergeleitet, MVMs werden ordnungsgemäß beendet und die zugrunde liegende Worker-Instance wird beendet. Lambda überwacht und alarmiert kontinuierlich die Lebenszyklusaktivitäten seiner Flotte.

Die gesamte Datenebenenkommunikation mit den Workern wird mit Advanced Encryption Standard mit Galois/Counter Mode (AES-GCM) verschlüsselt. Abgesehen von Operationen auf Datenebene können Kunden nicht direkt mit einem Worker interagieren, da dieser in einer netzwerkisolierten Amazon-VPC gehostet wird, die von Lambda in den Servicekonten von Lambda verwaltet wird.

Wenn ein Worker eine neue Ausführungsumgebung erstellen muss, erhält er eine zeitlich begrenzte Berechtigung, auf Artefakte der Kundenfunktion zuzugreifen. Diese Artefakte sind speziell für die Ausführungsumgebung und die Worker von Lambda optimiert. Funktionscode, der im ZIP-Format hochgeladen wird, wird einmal optimiert und dann in einem verschlüsselten Format mit einem von AWS verwalteten Schlüssel und AES-GCM gespeichert.

Funktionen, die mit dem Container-Image-Format auf Lambda hochgeladen wurden, werden ebenfalls optimiert. Das Container-Image wird zuerst von seiner ursprünglichen Quelle heruntergeladen, in verschiedene Chunks optimiert und dann mit einer authentifizierten konvergenten Verschlüsselungsmethode, die eine Kombination aus AES-CTR, AES-GCM und SHA-256 MAC verwendet, als verschlüsselte Chunks gespeichert. Mit der konvergenten Verschlüsselungsmethode kann Lambda verschlüsselte Chunks sicher deduplizieren. Alle Schlüssel, die zur Entschlüsselung von Kundendaten erforderlich sind, werden durch vom Kunden verwaltete AWS KMS Customer Master Key (CMK) geschützt. Die CMK-Nutzung durch den Lambda-Service steht Kunden in AWS CloudTrail-Protokollen zur Nachverfolgung und Prüfung zur Verfügung.

Technologien zur Lambda-Isolierung

Lambda verwendet eine Vielzahl von Open-Source- und proprietären Isolationstechnologien zum Schutz von Workern und Ausführungsumgebungen. Jede Ausführungsumgebung enthält eine dedizierte Kopie der folgenden Elemente:

- Den Code der jeweiligen Funktionsversion.
- Alle AWS-Lambda-Ebenen, die für Ihre Funktionsversion ausgewählt wurden.
- Die gewählte Funktionslaufzeit (z. B. Java 11, NodeJS 12, Python 3.8 usw.) oder die benutzerdefinierte Laufzeit der Funktion.
- Ein beschreibbares /tmp-Verzeichnis.
- Einen minimalen Linux-Benutzerraum basierend auf Amazon Linux 2.

Ausführungsumgebungen werden durch mehrere Container-ähnliche Technologien, die in den Linux-Kernel integriert sind, zusammen mit proprietären AWS-Isolationstechnologien voneinander isoliert. Zu diesen Technologien gehören:

- cgroups – Wird verwendet, um den Zugriff der Funktion auf CPU und Speicher zu beschränken.
- Namespaces – Jede Ausführungsumgebung wird in einem dedizierten Namespace ausgeführt. Dazu haben wir eindeutige Gruppen-Prozess-IDs, Benutzer-IDs, Netzwerkschnittstellen und andere Ressourcen, die vom Linux-Kernel verwaltet werden.
- seccomp-bpf – Beschränkt die Systemaufrufe (Syscalls), die innerhalb der Ausführungsumgebung verwendet werden können.
- iptables und Routingtabellen – Zum Verhindern von eingehenden Netzwerkkommunikation und dem Isolieren von Netzwerkverbindungen zwischen MVMs.
- chroot – Bietet Bereichszugriff auf das zugrunde liegende Dateisystem.
- Firecracker-Konfiguration – Wird verwendet, um den Durchsatz von Blockgeräten und Netzwerkgeräten zu begrenzen.
- Firecracker-Sicherheitsfunktionen – Weitere Informationen zum aktuellen Sicherheitsdesign von Firecracker finden Sie im neuesten Designdokument von Firecracker.

Zusammen mit den proprietären AWS-Isolationstechnologien bieten diese Mechanismen eine starke Isolierung zwischen Ausführungsumgebungen.

Speicherung und Status

Ausführungsumgebungen werden niemals über verschiedene Funktionsversionen oder Kunden hinweg wiederverwendet, aber eine einzelne Umgebung kann von Aufrufen derselben Funktionsversion wiederverwendet werden. Das bedeutet, dass Daten und Status zwischen Aufrufen bestehen bleiben können. Daten und/oder Status können stundenlang bestehen bleiben, bevor sie im Rahmen des normalen Lebenszyklusmanagements der Ausführungsumgebung zerstört werden. Aus Leistungsgründen können Funktionen dieses Verhalten nutzen, um die Effizienz zu verbessern, indem sie lokale Caches oder langlebige Verbindungen zwischen Aufrufen behalten und wiederverwenden. In einer Ausführungsumgebung werden diese Mehrfachaufrufe von einem einzigen Prozess verarbeitet, sodass jeder prozessweite Status (z. B. ein statischer Zustand in Java) für zukünftige Aufrufe zur Wiederverwendung verfügbar sein kann, wenn der Aufruf in einer wiederverwendeten Ausführungsumgebung erfolgt.

Jede Lambda-Ausführungsumgebung enthält auch ein beschreibbares Dateisystem, verfügbar unter `/tmp`. Dieser Speicher ist nicht über Ausführungsumgebungen hinweg zugänglich oder wird nicht gemeinsam genutzt. Wie beim Prozessstatus bleiben die Daten, die in `/tmp` geschrieben werden, für die gesamte Lebensdauer der Ausführungsumgebung erhalten. Auf diese Weise können teure Übertragungsvorgänge, wie das Herunterladen von Modellen für Machine Learning (ML), über mehrere Aufrufe hinweg amortisiert werden. Funktionen, die keine Daten zwischen Aufrufen beibehalten wollen, sollten entweder nicht in /tmp schreiben oder ihre Dateien zwischen Aufrufen aus /tmp löschen. Das `/tmp`-Verzeichnis wird von einem Amazon-EC2-Instance-Speicher unterstützt und im Speicher verschlüsselt.

Kunden, die Daten außerhalb der Ausführungsumgebung im Dateisystem speichern möchten, sollten die Lambda-Integration mit dem Amazon Elastic File System (Amazon EFS) in Erwägung ziehen. Weitere Informationen finden Sie unter Verwenden von Amazon EFS mit AWS Lambda.

Wenn Kunden Daten oder Status nicht über Aufrufe hinweg behalten möchten, empfiehlt Lambda, den Ausführungskontext oder die Ausführungsumgebung nicht zum Speichern von Daten oder Status zu verwenden. Wenn Kunden aktiv verhindern möchten, dass Daten oder Status über Aufrufe hinweg verloren gehen, empfiehlt Lambda, für jeden Status eigene Funktionen zu erstellen. Lambda empfiehlt Kunden nicht, den sicherheitsempfindlichen Status in der Ausführungsumgebung zu verwenden oder zu speichern, da er zwischen Aufrufen mutieren kann. Wir empfehlen stattdessen, den Status bei jedem Aufruf neu zu berechnen.

Laufzeitwartung in Lambda

Lambda unterstützt diese Laufzeiten, indem es kontinuierlich nach kompatiblen Updates und Sicherheits-Patches sucht und diese bereitstellt sowie andere Laufzeitwartungsaktivitäten durchführt. Auf diese Weise können sich Kunden nur auf die Wartung und Sicherheit von Code konzentrieren, der in ihrer Funktion und Ebene enthalten ist. Das Lambda-Team verwendet Amazon Inspector, um bekannte Sicherheitsprobleme sowie andere benutzerdefinierte Benachrichtigungsmechanismen für Sicherheitsprobleme und Listen zur Vorabveröffentlichung zu ermitteln, um sicherzustellen, dass unsere Laufzeitsprachen und die Ausführungsumgebung gepatcht bleiben. Wenn neue Patches oder Updates identifiziert werden, testet und stellt Lambda die Updates der Laufzeit ohne Beteiligung der Kunden bereit. Weitere Informationen zum Compliance-Programm von Lambda finden Sie im Abschnitt „Lambda und Compliance“ dieses Dokuments.

In der Regel ist keine Aktion erforderlich, um die neuesten Patches für unterstützte Lambda-Laufzeiten abzurufen, aber manchmal sind Aktionen erforderlich, um Patches vor ihrer Bereitstellung zu testen (z. B. bekannte inkompatible Laufzeit-Patches). Wenn eine Handlung von Kunden erforderlich ist, kontaktiert Lambda sie über das Personal Health Dashboard über die E-Mail im AWS-Konto oder auf andere Weise mit den spezifischen Aktionen, die ergriffen werden müssen.

Kunden können in Lambda andere Programmiersprachen verwenden, indem sie eine benutzerdefinierte Laufzeit implementieren. Für benutzerdefinierte Laufzeiten liegt die Wartung der Laufzeit in der Verantwortung des Kunden, einschließlich der Sicherstellung, dass die benutzerdefinierte Laufzeit die neuesten Sicherheits-Patches enthält. Weitere Informationen finden Sie unter Benutzerdefinierte AWS-Lambda-Laufzeiten im AWS-Lambda-Entwicklerhandbuch.

Wenn Betreuer der Upstream-Laufzeitsprache ihre Sprache als End-Of-Life (EOL) markieren, berücksichtigt Lambda dies, indem es die Sprachversion der Laufzeit nicht mehr unterstützt. Wenn Laufzeitversionen in Lambda als veraltet markiert sind, unterstützt Lambda nicht länger die Erstellung neuer Funktionen und Updates zu vorhandenen Funktionen, die in der veralteten Laufzeit erstellt wurden. Um Kunden über das bevorstehende Veralten von Laufzeiten zu informieren, sendet Lambda Benachrichtigungen an Kunden über das bevorstehende Veraltungsdatum und darüber, was sie erwarten können. Lambda stellt keine Sicherheitsaktualisierungen, keinen technischen Support und keine Hotfixes für veraltete Laufzeiten bereit und behält sich das Recht vor, jederzeit Aufrufe von Funktionen zu deaktivieren, die so konfiguriert sind, dass sie mit einer veralteten Laufzeit ausgeführt werden. Wenn Kunden weiterhin veraltete oder nicht unterstützte Laufzeitversionen ausführen möchten, können sie ihre eigene benutzerdefinierte AWS Lambda-Laufzeitumgebung

erstellen. Einzelheiten zur Deprekation von Laufzeiten finden Sie in der Laufzeit-Supportrichtlinie für AWS Lambda.

Überwachung und Prüfung von Lambda-Funktionen

Sie können Lambda-Funktionen mit vielen AWS-Services und -Methoden überwachen und prüfen, einschließlich der folgenden Services.

Amazon CloudWatch

AWS Lambda überwacht automatisch Lambda-Funktionen in Ihrem Namen. Über Amazon CloudWatch werden Metriken wie die Anzahl der Anforderungen, die Ausführungsdauer pro Anforderung und die Anzahl der Anfragen, die zu einem Fehler geführt haben, gemeldet. Diese Metriken werden auf Funktionsebene verfügbar gemacht, die Sie dann nutzen können, um CloudWatch-Alarme festzulegen. Eine Liste der von Lambda bereitgestellten Metriken finden Sie unter AWS LambdaMetriken.

Amazon CloudTrail

Mithilfe von AWS CloudTrail können Sie Governance, Compliance, Betriebs- und Risikoprüfung Ihres gesamten AWS-Kontos, einschließlich Lambda, implementieren. Mit CloudTrail können Sie Kontoaktivitäten im Zusammenhang mit Aktionen in Ihrer AWS-Infrastruktur protokollieren, kontinuierlich überwachen und speichern. So erhalten Sie einen vollständigen Ereignisverlauf der Aktionen, die über die AWS Management Console, AWS SDKs, Befehlszeilen-Tools und andere AWS-Services erfolgen. Mit CloudTrail können Sie die Protokolldateien optional verschlüsseln mithilfe von AWS KMS und auch die CloudTrail-Protokolldatei-Integritätsprüfung für eine positive Bestätigung zu nutzen.

AWS X-Ray

Mithilfe von AWS X-Ray können Sie die Produktion und verteilte Lambda-basierte Anwendungen analysieren und debuggen, wodurch Sie die Leistung Ihrer Anwendung und der zugrunde liegenden Services verstehen und so die Hauptursache von Leistungsproblemen identifizieren und Fehler beheben können. Die End-to-End-Ansicht von X-Ray von Anforderungen, die Ihre Anwendung durchlaufen, zeigt eine Übersicht der zugrunde liegenden Komponenten der Anwendung, sodass Sie Anwendungen während der Entwicklung und in der Produktion analysieren können.

AWS Config

Mithilfe von AWS Config können Sie Konfigurationsänderungen an den Lambda-Funktionen (einschließlich gelöschter Funktionen), Laufzeitumgebungen, Tags, Handler-Namen, Codegröße, Speicherzuweisung, Timeout-Einstellungen und Parallelitätseinstellungen sowie Zuweisungen zur Lambda-IAM-Ausführungsrolle, Subnetz und Sicherheitsgruppen nachverfolgen. Auf diese Weise erhalten Sie einen ganzheitlichen Überblick über den Lebenszyklus der Lambda-Funktion und können diese Daten für potenzielle Prüfungs- und Compliance-Anforderungen bereitstellen.

Architektur und Betrieb von Lambda-Funktionen

In diesem Abschnitt werden die Architektur und der Betrieb von Lambda behandelt. Informationen zu standardmäßigen bewährten Methoden für Serverless-Anwendungen finden Sie im Whitepaper Serverless Applications Lens, in dem die Säulen des AWS Well Architected Framework in einem Serverless-Kontext definiert und untersucht werden.

- Die Säule „Operative Exzellenz“ – Umfasst die Fähigkeit, Systeme auszuführen und zu überwachen, einen geschäftlichen Nutzen zu erbringen und unterstützende Prozesse und Verfahren kontinuierlich zu verbessern.
- Die Säule „Sicherheit“ – Die Fähigkeit, Informationen, Systeme und Ressourcen mithilfe von Risikobewertungen und Migrationsstrategien zu schützen und gleichzeitig einen Mehrwert für das Geschäft zu liefern
- Die Säule „Zuverlässigkeit“ – Die Fähigkeit eines Systems, sich von Infrastruktur- oder Serviceunterbrechungen zu erholen, Datenverarbeitungsressourcen je nach Bedarf dynamisch anzufordern und Unterbrechungen, z. B. aufgrund von Fehlkonfigurationen oder vorübergehenden Netzwerkproblemen, zu minimieren.
- Die Säule „Leistungseffizienz“ – Die effiziente Nutzung von Rechenressourcen entsprechend den Anforderungen sowie der Aufrechterhaltung dieser Effizienz, während sich die Nachfrage ändert und die Technologien weiterentwickeln.
- Die Säule „Kostenoptimierung“ – Der kontinuierliche Prozess der Verfeinerung und Verbesserung, um sicherzustellen, dass Geschäftsergebnisse erzielt werden, während die Kosten bei sich ändernden Anforderungen und Technologien minimiert werden.

Das Whitepaper Serverless Applications Lens enthält Themen wie Protokollierung von Metriken und Alarmierung, Drosselung und Grenzwerte, Zuweisen von Berechtigungen für Lambda-Funktionen und Bereitstellen sensibler Daten für Lambda-Funktionen.

Lambda und Compliance

Wie im Abschnitt „Modell der geteilten Verantwortung“ erwähnt, sind Sie dafür verantwortlich, zu bestimmen, welche Compliance-Regelung für Ihre Daten gilt. Nachdem Sie die Anforderungen Ihres Compliance-Regimes festgelegt haben, können Sie die verschiedenen Lambda-Funktionen verwenden, die diesen Steuerelementen entsprechen. Sie können sich an AWS-Experten (wie Solution Architects, Fachexperten, technische Kundenbetreuer und andere Personalressourcen) wenden, um Unterstützung zu erhalten. AWS kann Kunden jedoch nicht darüber beraten, ob (oder welche) Compliance-Regelungen für einen bestimmten Anwendungsfall gelten.

Seit November 2020 ist Lambda für SOC 1-, SOC 2- und SOC 3-Berichte vorgesehen, bei denen es sich um Prüfungsberichte von unabhängigen Drittanbietern handelt, die zeigen, wie AWS wichtige Compliance-Kontrollen und -Ziele erreicht. Eine aktuelle Liste der Compliance-Informationen finden Sie auf der Seite Im Rahmen des Compliance-Programms zugelassene AWS-Services.

Aufgrund der Sensibilität einiger Compliance-Berichte können sie nicht öffentlich geteilt werden. Um auf diese Berichte zuzugreifen, können Sie sich beim AWS Management Console anmelden und AWS Artifact verwenden – einem kostenlosen Self-Service-Portal für den bedarfsorientierten Abruf von AWS-Compliance-Berichten.

Lambda-Ereignisquellen

Lambda lässt sich über die direkte Integration und den Amazon-EventBridge-Ereignisbus in mehr als 140 AWS-Services integrieren. Die am häufigsten verwendeten Lambda-Ereignisquellen sind:

- Amazon API Gateway
- Amazon CloudWatch Events
- Amazon CloudWatch Logs
- Amazon DynamoDB-Streams
- Amazon EventBridge
- Amazon Kinesis Data Streams
- Amazon S3
- Amazon SNS
- Amazon SQS
- AWS Step Functions

Mit diesen Ereignisquellen können Sie:

- Die Option AWS Identity and Access Management verwenden, um Zugriff auf den Service und die Ressourcen sicher zu verwalten.

- Gespeicherte Daten verschlüsseln.* Alle Dienste verschlüsseln Daten während der Übertragung.
- Über Ihre Amazon Virtual Private Cloud mithilfe von VPC-Endpunkten (unterstützt von AWS PrivateLink) zugreifen.
- Amazon CloudWatch Application Insights verwenden, um Metriken zu erfassen und darüber Berichte sowie Alarme zu erstellen.
- Die Option AWS CloudTrail verwenden, um Kontoaktivitäten im Zusammenhang mit Aktionen in Ihrer AWS-Infrastruktur zu protokollieren, kontinuierlich zu überwachen und zu speichern. So erhalten Sie einen vollständigen Ereignisverlauf der Aktionen, die über die AWS Management Console >AWS SDKs, Befehlszeilen-Tools und andere AWS-Services erfolgen.

*Zum Zeitpunkt der Veröffentlichung war die Verschlüsselung von gespeicherten Daten für Amazon EventBridge nicht verfügbar. Sehen Sie regelmäßig auf den Homepages des Services nach, um Updates zu diesen Funktionen zu erhalten.

Fazit

AWS Lambda bietet ein leistungsstarkes Toolkit zum Erstellen sicherer und skalierbarer Anwendungen. Viele der bewährten Methoden für Sicherheit und Compliance in Lambda sind dieselben wie in allen AWS-Services – einige sind jedoch spezifisch für Lambda. In diesem Whitepaper werden die Vorteile von Lambda, dessen Eignung für Anwendungen und die von Lambda verwaltete Laufzeitumgebung beschrieben. Es enthält auch Informationen zur Überwachung und Prüfung sowie bewährte Methoden für Sicherheit und Compliance. Wenn Sie über Ihre nächste Implementierung nachdenken, beachten Sie, was Sie über AWS Lambda gelernt haben und wie Ihre nächste Workload-Lösung dadurch verbessert werden kann.

Mitwirkende

An diesem Dokument haben folgende Personen mitgewirkt:

- Mayank Thakkar, Global Life Sciences Solutions Architect
- Marc Brooker, Senior Principal Engineer (Serverless)
- Osman Surkatty, Senior Security Engineer (Serverless)

Weitere Informationen

Weitere Informationen finden Sie unter:

- Das Modell der geteilten Verantwortung, das den Standpunkt von AWS zu Sicherheit allgemein erklärt.
- Bewährte Methoden von AWS für die Sicherheit, die Empfehlungen für den AWS IAM (Identity and Access Management)-Service abdecken.
- Die Serverless Applications Lens deckt das AWS Well-Architected Framework ab und identifiziert Schlüsselelemente, um sicherzustellen, dass Ihre Workloads gemäß bewährten Methoden konzipiert werden.
- Die Einführung in die AWS-Sicherheit bietet eine umfassende Einführung in das Denken über die Sicherheit in AWS.
- AWS – Risiko und Compliance bietet einen Überblick über die Compliance in AWS.

Dokumentversionen

Abonnieren Sie den RSS-Feed, um über Aktualisierungen des Whitepapers benachrichtigt zu werden.

Update-Historie-Änderung	Update-Historie-Beschreibung	Update-Historie-Datum
Aktualisiert	Bedeutende Aktualisierungen	15. Februar 2021
Erste Veröffentlichung	Erstveröffentlichung des Whitepapers	3. Januar 2019

Hinweise

Kunden sind eigenverantwortlich für die unabhängige Bewertung der Informationen in diesem Dokument zuständig. Dieses Dokument: (a) dient rein zu Informationszwecken, (b) spiegelt die aktuellen Produktangebote und Verfahren von AWS wider, die sich ohne vorherige Mitteilung ändern können, und (c) impliziert keinerlei Verpflichtungen oder Zusicherungen seitens AWS und dessen Tochtergesellschaften, Lieferanten oder Lizenzgebern. AWS-Produkte oder -Services werden im vorliegenden Zustand und ohne ausdrückliche oder stillschweigende Gewährleistungen, Zusicherungen oder Bedingungen bereitgestellt. Die Verantwortung und Haftung von AWS gegenüber seinen Kunden wird durch AWS-Vereinbarungen geregelt. Dieses Dokument ist weder ganz noch teilweise Teil der Vereinbarungen zwischen AWS und seinen Kunden und ändert diese Vereinbarungen auch nicht.